The Official AMOS CLUB

NEWSLETTER Vol2 .ISSUE 1

☞ Your AMOS CLUB Membership Number

2 AP0315

The year starts here ! Welcome to the first newsletter of subscription year 2, just as Francois is starting work on AMOS 2 ! The new version will have all the power of AMOS, with the best points from Easy AMOS' editor facilities .

The good news is that all your old programs and extensions will work with AMOS 2, the bad news is that it won't be available as a P.D upgrade, although an upgrade service is planned (you send in your old AMOS and an upgrade fee.), as AMOS 2 is only in the "We've only just started writing it, we'll invent the prices later" stage, no price or release date is available yet, it looks like a possible October or November release ,This of course is pure guesswork !.

Kevin Hall We are saddened to report the death of Journalist Kevin Hall in a motorcycle accident. Kevin chaired the CIX AMOS Conference, as well as supporting AMOS in JAM magazine and reviewing AMOS software in the new Amigaware magazine. Kevin was a major supporter of AMOS, and his input will be greatly missed.

Our Apologies to some of our overseas members, unfortunately the British postal network fell apart over Christmas, with newsletters going astray all over the place, so many of the issue 5/6 newsletters had to be re-sent after the post had sorted itself out. ALL the special offers in issue 5/6 are now valid until Volume 2, Issue 2 (The next newsletter), and we are running another competition for all those who didn't have time to enter the first one.

Easy AMOS has had its release date set for April 23rd, although Europress haven't stated what year yet ! This is the long awaited beginners version of AMOS with an easier editor and tutorial manual. The price seems to be around £35.00 and there is no news yet on whether there will be a discount for existing AMOS owners (in our opinion there should be !).

Good news also for Amiga Plus owners, if you have an older version of AMOS 3D, where the Object Modeller won't work on your machine, send it back to Europress/

Inside this issue:

New Members 3
Checksum Update 3
Absolute Beginners (Logic
and Zones) 4
THE DISK 8
The Kennard Effect (Real Time Lens
and Shadows) 9
The Vault (short but useful routines) .. 12
10 Liners (Othello, Card Factory
and Find the Lady) 14
Using the Serial Extension 17
Licenseware Reviewed 19
Book Report 19
Special Offers 20

Mandarin, and they'll replace it with the new version which does work on the plus. It was also found that one or two of the older AMOS P.D programs (using RAMOS 1.1) have problems on the plus. This is solved by using the programs with AMOS 1.3 or above, and either compiling them, or changing the RAMOS on the disk to RAMOS 1.3, Sandra has now converted most of the older disks to AMOS 1.3, thus solving the problem.

Good news also for you lazy types ! If you don't want to type in any of the listings from this newsletter, or you want graphics files ready to use with them, we have produced **THE DISK**. This is the AMOS Newsletter on a disk with all the programs ready to run, including all the required graphics and sound, as well as extra useful programs and some P.D. **THE DISK** will be produced for each newsletter and is only available to AMOS Club members. See page 8 for details.

Extension Information Good news for CDTV developers, a Shareware CDTV/ A690 controller extension allowing you to control the music CD playing from within AMOS is about to be released ! This will be extension number 11.

Down to some serious grovelling ! I mistakenly credited the X-Stitch program in last issues review to Derek Pell, when it was in fact written by Jeff Tullin. Sorry Jeff !

Serial Extension fix If you have been trying to use the Serial extension and find that the machine locks out after you Serial Open the second time, simply copy the **SERIAL.DEVICE** from the Devs directory of your AMOS 1.23 or 1.34 update disk (Not 1.3 -1.33) into the Devs directory of your boot disk, as the older Workbench serial device has a small bug. Check out our article on programming with the serial extension on page 17.

A word to anyone who uses the AMOS CIX conference, where certain people have been claiming that Deja Vu Licensed software is just a way of selling P.D at high prices. Please note that Licensed Software is **COMMERCIAL** Software sold at a very low price (half the price of budget software) and is a very good way of selling software that gets an excellent royalty return for the author, much better than most software companies do !

If the disks were P.D (of any sort) then neither myself nor most of the Licenseware programmers would have released anything on it ! Most of the people trying to put down Licenseware are either trying to force Shareware to work (It won't work in most cases, except for a very few programs that either require a huge manual, or are upgraded constantly) or want all software to be free and all programmers to work for nothing. Which is fine, the first P.D company to offer to feed me, pay all my bills, keep a decent roof over my head and provide for the future of my family and guarantee the same for all the other programmers, will get all the free software it wants.

Honest Person. John, from Orbit Software, one of the Licenseware dealers was offered a pirated version of Dogfight II from a dodgy type at one of the recent shows. So he trashed it (They don't make disk drives like they used to) and sold the dodgy type a genuine one ! Programmers 1 - Pirates 0 !

Another Honest Person. Sandra recently had a letter from a customer who had loaned one of his Licenseware disks to a friend. Suspecting that his friend had made himself a copy, he sent Sandra the payment for his friend's License fee. Programmers 2 -Pirates 0 !

Checksum Modification The Checksum checker in issue 5/6 has gone down quite well, but I was given a much improved version by Dr L Rowe about 3 days after he saw it. Basically, all you need to replace is the final Checksum subroutine with the improved one, and you will get the following advantages.

1) It will completely ignore leading spaces in the line.

2) It will spot the most common typing error, the transposition of two adjacent letters, e.g PIRNT instead of PRINT. All checksummed programs from this issue onwards will use the new routine, so dig up your version of it from the last issue and erase the **CHECKSUM:** subroutine, replacing it with this one... If you notice a way of seriously improving any of the programs we print, please tell us, we're quite happy to print updates !

```
1) CHECKSUM : [ 54]
2) L2$ = L$ [ 8]
3) Rem remove spaces [ 74]
4) A = 1 [ 77]
5) While Mid $ ( L $ , A , 1 ) = "
 " : Inc A : Wend [ 37]
6) L $ = Upper $ ( L $ ) [ 91]
7) C = 0 : X = 1 [ 33]
8) For B = A To Len ( L $ ) [ 99]
9) C = C xor Asc ( Mid $ ( L $ ,
 B , 1 ) ) xor X [ 75]
10) Inc X [ 61]
11) Next B [ 98]
12) Print " ( " ; L ; " ) = " ; C [
 106]
13) Print # 2 , Str $ ( L ) + " ) "
 + L2 $ + " [ " + Str $ ( C ) + "
 ] " [ 54]
14) Return [ 42]
```

The Checksum Checker program is on this issue of THE DISK (Page 8).

New Members

If this is your first AMOS Newsletter, welcome to the AMOS Club ! Your membership number is in the box on the front page, please quote this when using the helpline and when replying to special offers. Your membership is now valid for Volume 2, Issues 1-6 of the newsletter.

If you joined very late in the membership year, apart from receiving issues 1-6 at the same time, your membership will be valid for a fixed time past the resubscription date. This will be shown with your membership number .

Normally newsletters are every 2 months, although gaps are sometimes longer if no one is sending in articles !

When typing in listings from the newsletter, don't enter the Number and bracket at the start of the line or the number in square brackets at the end, as these are only used for checking your program with the checksum checker from issue 5/6 .

If all else fails, you can always phone !

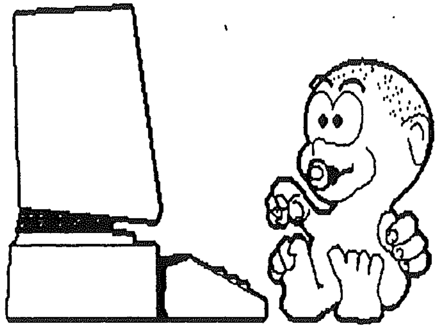
The AMOS Helpline

As an AMOS Club member, you can call the AMOS Helpline at Shadow Software any time between 2p.m and 7p.m if you need help with your AMOS programming. The number is

0271-23544 (Shadow Software)

remember to have your newsletter handy to quote your membership number.

Absolute Beginners



Logically Speaking

I could introduce this subject with an impersonation of a certain pointy eared science officer, but I don't want to get beaten up by a bunch of annoyed treckies, so I'll avoid the usual daft intro and get right into the educational bit !

This article will explain the use of Boolean Logic functions in AMOS, and Basic in general (A warning to our younger readers, don't say Boolean Logic Functions anywhere near a lesser intelligent adult, or you'll probably get a clout around the ears). Despite the complicated name, these functions are only the AND, OR, NOT and XOR functions, which are simple to use, and very useful for comparing events.

For instance, say you have variable A and variable B, and you want to warn the user if either A, or B is greater than 100 (Say for a temperature monitor). You could do:

```
If A>100
If B>100
Print "Temperature Warning !"
Endif
Endif
```

but you could also do:

```
If A>100 Or B>100
Print "Temperature Warning !"
Endif
```

Which involves less programming and is slightly faster. The **Or** function compares the two numbers presented to it (in this case the True or False of the two greater than checks) and returns True if either of the inputs are True. So if Input 1 **OR** Input 2 is true, then the answer will be true.

This can be shown in a logic table:

Input 1	**OR**	**Input 2**	**Result**
False	**OR**	**False**	**False**
True	**OR**	**False**	**True**
False	**OR**	**True**	**True**
True	**OR**	**True**	**True**

The **AND** Function works in a similar way, but to get a true result, Both Input 1 **AND** Input 2 must be true. The logic table below shows the results:

Input 1	**AND**	**Input2**	**Result**
False	**AND**	**False**	**False**
True	**AND**	**False**	**False**
False	**AND**	**True**	**False**
True	**AND**	**True**	**True**

The **XOR** (or Exclusive Or) function returns true if Input 1 is true or Input 2 is true, but **NOT** if Both are true. E.g

Input 1	**XOR**	**Input2**	**Result**
False	**XOR**	**False**	**False**
True	**XOR**	**False**	**True**
False	**XOR**	**True**	**True**
True	**XOR**	**True**	**False**

NOT simply reverses a result, so NOT True would be False and NOT False would be True. e.g

```
If NOT A>100
Print "A is not >100"
Endif
```

In AMOS, these functions work on more than true or false. If used with normal variables, they work on the individual bits of the variables. So if you did **A=52 B=37 Print A And B** you would get a result of 36, as the individual bits of 52 (%11010) and 37 (%11011) are compared, and only the 32 and 4 bits are both set, giving you 32. Try the following tester program out:

```
1) ' Logic Demonstrator [ 88]
2) ' Aaron Fothergill [ 88]
3) ' For the AMOS Club [ 45]
4) ' [ 38]
5) Input " Enter A Number " ; A [
 30]
6) Print " A = " ; Bin $ ( A ) ; " in
 Binary " [ 36]
7) Input " Enter another Number " ;
 B [ 16]
8) Print " B = " ; Bin $ ( B ) ; " in
 Binary " [ 36]
9) Print [ 118]
10) Print " A OR B = " ; Bin $ ( A
 or B ) ; " = " ; ( A or B ) [ 52]
11) Print [ 118]
12) Print " A AND B = " ; Bin $ (
 A and B ) ; " = " ; ( A and B ) [
 83]
13) Print [ 118]
14) Print " A XOR B = " ; Bin $ (
 A xor B ) ; " = " ; ( A xor B ) [
 93]
15) Print [ 118]
```

TOME TRICK

If you are an AMOS TOME user (i.e just about everybody), try pressing Shift ~ when you are using the TOME editor and have the TOME disk in the drive. It just goes to show that the TOME editor is all singing and dancing !

AMOS Zones

This absolute beginners article crosses into the area of the Learners series, as it deals with an element of AMOS that isn't in normal Basic, that of the ZONE commands.

In older basics, the only way to check if an object was in a box was to do an If statement like the following:

```
If ObjectX > = BoxX and ObjectX <
 = BoxX + Width and ObjectY > =
 BoxY and ObjectY < = BoxY +
 Height Then Print "In the Box"
```

Which for several boxes (or buttons) was really nasty ! In AMOS, you simply set the zones you want to check for using the Reserve Zone and Set Zone commands, and you can then use the **=Mouse Zone** or **=Zone(X,Y)** functions to tell you what zone the Mouse Pointer and the Co-ordinates X,Y are in. For Instance:

```
Reserve Zone 2
Set Zone 1,0,0 to 160,199
Set Zone 2,160,0 to 319,199
Do
Home
Print Mouse Zone
Loop
```

This will define two zones, one in the left half of the screen, and the other in the right. And tell you which zone the mouse is in.

A tip when setting up zones is to use the box command to draw a box at the same co-ordinates as the zone, thus showing you where you defined your zone. This allows you to troubleshoot more easily.

Use of the Zone Commands. First of all, you have to use the **Reserve Zone n** command to tell AMOS how many zones you want to use. Please note that the number of zones is stored independantly for each screen. So you can have different zones on each screen, but you must remember to use reserve zone for each screen seperately. So if you want 8 zones, you do **Reserve Zone 8.**

Next you set the areas you want as zones with the **Set Zone** command. This requires a zone number (the number returned by **=Zone** or **=Mouse Zone**) and the co-ordinates (Top Left and Bottom Right) of the zone to be set up. e.g **Set Zone 1,TLX,TLY to BRX,BRY** Amos will now remember this zone.

A Set of Buttons. One of the more useful uses of Zones, is for selectors. Normally, a selector will wait for the user to press the mouse button, at which point, it looks at which button was pressed. What you normally do is check the Mouse key with the **=Mouse Key** function, to see if it has been pressed. If so, get **Mouse Zone** and see if it is the button you want. Have a go of the following couple of examples.

Remember not to type in the Line numbers or the checksum in square brackets at the end of the line, these are only for later checking with the Checksum Checker program .

```
1) ' Using Zones in AMOS [ 58]
2) ' Absolute Beginners ish 7 [ 56]
3) ' Aaron Fothergill [ 88]
4) ' [ 38]
5) ' A simple Mouse driven
 xylophone [ 84]
6) ' [ 38]
7) Screen Open 0 , 320 , 200 , 4 ,
 Lowres : Curs Off : Flash Off :
 Cls 0 [ 63]
8) Fade 1 , $0 , $FFF , $F , $F00 [
 118]
9) Dim N(7) : Rem 8 notes (0-7) [
 20]
10) Rem read in the note numbers
 [ 87]
11) For A=0 To 7 [ 23]
12) Read N(A) [ 61]
13) Next A [ 97]
14) Data 60 , 62 , 64 , 65 , 67 , 69
 , 70 , 72 [ 15]
15) Rem set up the keys [ 64]
16) Reserve Zone 8 : Rem 1 zone
 for each key [ 34]
17) Ink 1,1 [ 64]
18) For A=0 To 7 [ 23]
19) Bar A * 32 + 16 , 16 To A *
 32 + 45 , 96 [ 120]
20) Set Zone A + 1 , A * 32 + 16
 , 16 To A * 32 + 45 , 96 [ 63]
21) Next A [ 97]
22) Home : Paper 0 : Pen 1 :
 Centre "The Mousofone" [ 34]
23) OK=0 [ 13]
24) C=1 : Rem channel number for
 next note [ 35]
25) While OK = 0 : Rem carries on
 this loop while OK = 0 [ 53]
26) K = Mouse Key [ 64]
27) If  K = 2  [ 108]
28) Rem if right button is pressed,
 [ 83]
29) Rem end loop by setting ok=1
 [ 77]
30) OK = 1  [ 12]
31) End If : Rem (If K=2) [ 125]
32) If K = 1 [ 111]
33) Z = Mouse Zone [ 20]
34) If Z > 0 [ 124]
35) N = N ( Z - 1 ) : Rem get the
 note played [ 35]
36) Play C , N , 0 [ 18]
37) Rem this little trick allows the
 mousofone [ 45]
38) Rem to play 4 notes at once [
 33]
39) C = C * 2 : If C = 16 : C = 1 :
 End If [ 70]
40) Rem wait until mouse key is
 released or different note is [ 73]
41) Rem hit [ 47]
42) While Mouse Key > 0 and
 Mouse Zone = Z [ 55]
43) Wend [ 57]
44) Rem ok, continue [ 82]
45) End If : Rem (If Z>0) [ 109]
46) End If : Rem (If K=1) [ 126]
47) Wend [ 57]
```

To use the mouseophone, simply click on the 8 tone bars on the screen. These will play the 8 notes (C–C) that the mouseophone is capable of.

Note the use of the C = C * 2 trick to enable 4 note pollphony (4 notes at once).

```
1) ' The Lift [ 73]
2) ' By Aaron Fothergill [ 66]
3) ' Simple Zones example [ 44]
4) ' [ 38]
5) ' [ 38]
6) Screen Open 0 , 320 , 200 , 16
 , Lowres : Curs Off : Flash Off :
 Cls 0 [ 50]
7) Colour 1 , $ FFF : Colour 2 , $
 F00 [ 56]
8) Reserve Zone 8 [ 105]
9) For A = 0 To 7 [ 23]
10) Set Zone A + 1 , 64 , 16 + A
 * 16 To 256 , 30 + A * 16 [ 32]
11) Ink 1 , 1 [ 64]
12) Bar 64 , 16 + A * 16 To 256 ,
 30 + A * 16 [ 92]
13) Ink 0 , 1 , 0 [ 92]
14) Text 132 , A * 16 + 26 , "
 Floor " + Str $ ( 8 - A ) [ 58]
15) Next A [ 97]
16) Rem draw lift and get it as a
 bob [ 80]
17) Ink 2 , 0 , 0 [ 95]
18) Box 0 , 0 To 31 , 15 [ 120]
19) Get Bob 1 , 0 , 0 To 32 , 16 [
 14]
20) Cls 0 , 0 , 0 To 32 , 16 [ 110]
21) FLOOR = 7 : Rem Top floor [
 33]
22) LY = 0 : Rem lift y co -
 ordinate [ 6]
23) Double Buffer [ 26]
24) Autoback 0 [ 31]
25) Do [ 43]
26) Rem draw cable [ 51]
27) Ink 1 [ 92]
28) Draw 32 , 0 To 32 , LY [ 13]
29) Ink 0 [ 93]
30) Draw 32 , LY To 32 , 199 [ 11]
31) Rem plot lift [ 75]
32) Bob 1 , 16 , LY , 1 [ 80]
33) Rem check if lift is at correct
 floor [ 44]
34) If LY < > ( 7 - FLOOR ) * 16
 + 16 [ 109]
35) Rem if Not , Then moveit
 towards it [ 66]
36) LY = LY + Sgn ( ( 7 - FLOOR
 ) * 16 + 16 - LY ) [ 54]
37) If LY = ( 7 - FLOOR ) * 16 +
 16 [ 68]
38) Bell [ 38]
39) End If [ 64]
40) End If [ 64]
41) Rem is mouse button pressed [
 81]
42) If Mouse Key > 0 [ 24]
43) Rem lift button pressed ? [ 35]
44) Z = Mouse Zone [ 20]
45) If Z > 0 [ 124]
```

```
Rem yes , so select new floor [ 40]
FLOOR = 8 - Z [ 43]
Bell [ 38]
End If [ 64]
End If [ 64]
Loop [ 61]
```

AMOS 1.34

AMOS has been upgraded again (That's one per newsletter so far !) and is now V1. 34 .

This version fixes many of the problems brought in with the Compiler (such as the VAL bug), and with the new V1.34 Compiler solves most of the problems in AMOS .

The AMOS Updater will hit P.D about the same time as you get this newsletter along with a new updater for the Compiler.

Contact Sandra at Deja Vu on **0942 495261** for details.

Upgrade as soon as possible, most of the bugs reported in on the helpline were fixed several versions ago !

The Disk

Cover disks have been a feature of many of the Amiga magazines over the last couple of years. They mean that listings don't have to be typed in, and graphics and sound can also be supplied to the reader. However, many magazines have lost readers because their price goes through the ceiling once the cost of a cover disk is added.

THE DISK is the optional AMOS Newsletter disk for AMOS Club members. As well as having all the listings of each newsletter ready typed, and all the required graphics, we will be adding even more, more 10-liners, more utilities, more examples, more graphics and it also means that we can put updates on the disk for the various AMOS utilities. (Issue 1 also has the AMOS Galaxy Rons game on it)

Each issue of the disk can be bought for **£1.75** (**£2.00** overseas) and is available direct from the AMOS Club when each issue of the newsletter is sent out. (Back issues are also available). Or if you want, we can send you each edition of THE DISK along with your newsletters as they come out, a subscription costs **£7** (**£10** overseas) and covers the six issues of THE DISK matching your newsletter subscription.

As THE DISK is only available to AMOS Club members, please use the order form on the back page and **remember to include your membership number.**

As with all offers from the AMOS Club, orders for the disk should be sent to

The AMOS Club (Dept TD), 1 Lower Moor, Whiddon Valley, Barnstaple, N.Devon. EX32 8NW. Making Cheques payable to THE AMOS Club .

THE KENNARD EFFECT

Real Time Lens and Shadowing Effects

A couple of months ago, I received a disk from Richard Kennard in Warwick, who was trying to write a game with a Predator like character who is invisible and can only be seen by the background distorting slightly around its shape.

Richard has managed to use a combination of a cutter mask (as used by Len Tucker in Jigmania to cut out jigsaw pieces) and an effect of the AMOS Bob Masking and screen bitplanes to create some pretty stunning effects.

Here follows Richard's principles translated from his demo (I would prefer ASCII text files for future articles though !)

o The Real Time Lens.

This effect allows you to move a bob around the screen, magnifying (or shifting) the background through the bob, rather like a magnifying glass. The principle is this..

1) If you have a bob with more colours than the screen, when that bob is placed on the screen, all the colours above the screen's maximum are placed on the screen as lower colours (basically because 1 or more bitplanes are not placed).

This means that if you have a 64 colour bob in colour 32 on a 32 colour screen, the bob will become colour 0. However, because only the real colour 0 on the bob is masked, this "fake" colour 0 won't be masked, and can be used to cut objects out. (This is how Jigmania cuts out its pieces). This means that if you design a 64 colour bob with the area you want to cut in colour 0 and the area you don't want in colour 32 you can paste this bob over a chunk of 32 colour screen, grab it as a bob (the colour 32 becomes 0, and is masked) and you get an irregular shaped bob cut out from a rectangular area.

2) If you screen copy a square chunk from your 32 colour background onto a hidden 32 colour screen, paste your "cutter" bob over it and grab it again (as a different bob), you can place this bob over your background. If you distort the square on the hidden screen in any way before placing the cutter bob over it, it will look as if the background is distorting through your bob. Zooming the area will cause a magnifying glass effect, random shifting will cause a water like effect.

Try the following example program (this requires the AMOS_Data disk) .

```
' Kennard Lens Principle Demonstrator [ 45]
Rem open a 64 colour screen and clear it to colour 32 [ 53]
Screen Open 1 , 320 , 33 , 64 , Lowres : Curs Off : Flash Off : Cls 32 [ 53]
Rem the lens can be any shape , as long as you do the see through bits [ 87]
Rem in colour 0 and the mask in colour 32 [ 76]
Gosub MAKELENS [ 117]
Rem change the above to MAKELENS2 to change the lens to a piece [ 53]
Rem of broken glass ! [ 38]
Rem reopen screen to 32 colours [ 52]
Screen Open 1 , 320 , 48 , 32
```

```
, Lowres : Curs Off : Flash Off : Cls 0 [ 53]
Rem load a 32 colour screen [ 56]
Load Iff " AMOS_DATA : IFF / AmosPic . IFF " , 0 [ 90]
Double Buffer [ 26]
Wait Vbl [ 82]
Limit Mouse [ 24]
Hide On [ 9]
Bob Update Off [ 49]
While Mouse Key = 0 [ 69]
Screen 0 [ 20]
Bob Clear [ 29]
Zoom Logic ( 0 ) , X , Y , X + 32 , Y + 32 To 1 , 0 , 0 , 40 , 40 [ 73]
Bob 1 , X , Y , 2 [ 65]
Bob Draw [ 78]
Screen Swap [ 21]
Wait Vbl [ 82]
Screen 1 [ 21]
Paste Bob 4 , 4 , 1 [ 45]
Get Bob 2 , 4 , 4 To 37 , 37 [ 11]
Screen 0 [ 20]
X = Max ( 0 , Min ( Screen Width - 32 , X Screen ( X Mouse ) ) ) [ 65]
Y = Max ( 0 , Min ( Screen Height - 32 , Y Screen ( Y Mouse ) ) ) [ 41]
Wend [ 57]
End [ 107]
MAKELENS : [ 45]
Rem draw a circle in colour 0 [ 55]
Ink 0 : For Q # = 0 To 2 . 1 * Pi # Step 0 . 02 [ 75]
Draw 16 , 16 To 16 + Cos ( Q # ) * 16 , 16 + Sin ( Q # ) * 16 [ 49]
Next Q # [ 85]
Ink 1 [ 92]
Circle 16 , 16 , 16 [ 53]
Rem grab it as our mask [ 63]
Get Bob 1 , 0 , 0 To 33 , 33 [ 8]
Return [ 42]
MAKELENS2 : [ 21]
Rem draw a funny lens [ 68]
Ink 0 [ 93]
Polygon 0 , 0 To 16 , 8 To 31 , 5 To 30 , 16 To 16 , 31 To 4 , 12 To 0 , 0 [ 66]
Ink 1 [ 92]
Polyline 0 , 0 To 16 , 8 To 31 , 5 To 30 , 16 To 16 , 31 To 4 , 12 To 0 , 0 [ 55]
Get Bob 1 , 0 , 0 To 33 , 33 [ 8]
Return [ 42]
```

Shadow Effect

The principle of colours having equivalents when there is a difference in numbers of bitplanes between the bob and the screen can be used the other way around for a Shading effect. If you put a 4 colour bob on a 64 colour screen, whenever the bob is over colours 0-3 it will be normal. When it is over colour 4, its colours will be 4-7, when it is over colour 8, they will be 8-11. If you define each set of 4 colours slightly darker, you do a wonderfull shadow effect. Have a go of the example on the following page.

```
' Kennard Shadow Principle demonstrator [ 62]
Rem open a 4 colour screen [ 63]
Screen Open 0 , 320 , 200 , 4 , Lowres : Curs Off : Flash Off : Cls 0 [ 63]
Locate 0 , 0 : Paper 2 : Pen 1 [ 14]
T $ = " Shadow Effect " [ 111]
Print T $ [ 9]
Rem get it as a 4 colour bob [ 48]
Get Bob 1 , 0 , 0 To Len ( T $ ) * 8 , 8 [ 31]
Rem reopen the screen to 64 colours [ 79]
Screen Open 0 , 320 , 200 , 64 , Lowres : Curs Off : Flash Off : Cls 0 [ 55]
Rem set colours [ 66]
Colour 1 , $ FFF : Colour 2 , $ F00 [ 56]
Rem set equivalent colours to darker shades [ 87]
For A = 1 To 7 [ 22]
Colour A * 4 , 0 [ 103]
Colour A * 4 + 1 , ( 15 - A ) * 256 + ( 15 - A ) * 16 + 15 - A [ 56]
Colour A * 4 + 2 , ( 15 - A ) * 256 [ 48]
Next A [ 97]
Rem draw shades [ 82]
For A = 1 To 15 [ 40]
Ink A * 4 [ 51]
Bar 64 + A * 16 , 0 To 79 + A * 16 , 199 [ 123]
Next A [ 97]
Hide On [ 9]
Rem show bob [ 55]
While Mouse Key = 0 [ 69]
Bob 1 , X Screen ( X Mouse ) , Y Screen ( Y Mouse ) , 1 [ 67]
Wait Vbl [ 82]
Wend [ 57]
```

So there you are, 2 pretty nifty effects, simply acheived in AMOS. All Richard asked for was a credit and for his name to be plastered in the newsletter. Well seing as he was the one who perfected the use of this effect and documented it properly, this effect from now on should be called the Kennard Effect.

TOME
Series IV

AMOS TOME is in the process of being updated to TOME Series IV, with new tricks and a more powerful extension.

The Good news, a full 80 page manual (in real paper !) as well as the hyper text manual now built into the TOME Editor, which now has its own programming language MaPLe (Map Programming Language).

The bad news is that the price has gone up by £5 to £24.99 (£29.99 for non members) to reflect the cost of the manual. (Although TOME V3.2 will still be available at the old price).

TOME Series IV is due for release around the end of March 92, and there will an upgrade service for existing TOME V3 owners, with a minimal upgrade fee to cover the cost of the new disk and manual.

So watch this space !

THE VAULT

This new section contains all those routines for particular tasks, that you keep forgetting !

- Toggle Variable T between 0 and 1

Assuming T is 0 or 1 to start with do:

```
T = 1 - T
```

If T is likely to be any other number do:

```
T = - ( T = 0 )
```

- Cycle Variable T from 0-N

```
T = ( T + 1 ) Mod ( N + 1 )
```

- Cycle Variable T through a complex sequence of n numbers (e.g 1,7,93,4,5 etc)

```
Dim T ( n ) : Rem set to number
 of numbers in sequence (in this
 case 5)
For A = 0 to n - 1 : Rem read in
 the numbers
Read T ( A )  : Next A
Data 1 , 7 , 93 , 4 , 5
P=0 : Rem P is the pointer to the
 position in the sequence
```

and now in your main loop, you can do

```
P = ( P + 1 ) Mod ( n + 1)
T = T ( P )
```

- Generate a random number of +-N,not including zero:

```
X = N * ( Rnd ( 1 ) * 2 - 1 )
```

- Remove spaces from the end of a string S$ (handy for when using dir first$)

```
While Right$ ( S$ , 1 ) = " " : rem
 Quote Space Quote
S$= Left$ ( S$ , len ( S$ ) - 1 )
Wend
```

- Get the File name from a Dir First$ or Dir Next$ string (Assumes no spaces in file name).

```
F$ = your string
F$ = Mid$ ( F$ , 2 )
F$ = Mid$ ( F$ , 1 , Instr ( F$ , "
 " ) - 1 )
```

- Read a Directory into an Array

```
Dim D$ ( 99 ) : Rem make larger
 for larger directories
F$ = Dir First$ ( " Df0 : " )
P=0 : While F$ < > "" : Rem
 Quote Quote
F$ = Mid$ ( F$ , 1 , instr ( F$ , "
 " , 2 ) - 1 )
Rem leave leading character as it
 shows whether the file is a file or
 a directory.
D$ ( P ) = F$:  Inc P
F$ = Dir Next$
Wend
Rem now display the directory
For A=0 to P-1 : Print D$(A) : Next
 A
```

You could also leave F$ in its full form (don't do the ≠Mid$ bit) so that you can store the filesize as well.

- Capitalize a Name N$ (Make first letter a capital, and all others Lower Case),

this also works for multiple word strings, making all first letters capital.

```
Line Input N$
N$=Lower$(N$)
P = 1 : S = 1
While P < = Len ( N$ )
If S = 1
N$ = Mid$ ( N$ , 1 , P - 1 ) +
 Upper$ ( Mid$ ( N$ , p , 1 ) ) +
 Mid$ ( N$ , P + 1 )
S=0 : Endif
If Mid$ ( N$ , P , 1 ) = " " : Rem
 Quote Space Quote
S = 1 : Endif
Inc P : Wend : Print N$
```

AMOS TOME

SHADOW SOFTWARE

"AMOS TOME is just one of those things you *have* to have if you're serious about AMOS." Phil South . Amiga Shopper March 1992

Now being used by **ALL the major AMOS programmers**, including Aaron Fothergill, Len Tucker (Sparx), Gary Symons, Stephen Hill, Peter Hickman and more. **AMOS TOME** has been upgraded to **V3.2**. It is compatible with the Compiler and AMOS 1.3 upwards. TOME now has **34 Commands**, and the editor is **still the most powerful map editor available on ANY computer.**

TOME has already been used to write 7 commercial games, and many more are being developed. 90% of games use map based backgrounds, making **AMOS TOME the most useful extension for AMOS.**

The TOME Goodies disk 1 is available to registered TOME users, with a **3D Dungeon game, 3D Isometric scrolling game and the source code to Magic Forest II**, a horizontally scrolling platform game, all written using AMOS TOME. **The TOME Goodies disk 2 will be available soon**, with 3 more game styles (including Hardware scrolling shoot-em-ups !).

Coming soon AMOS TOME Series IV, an upgrade from the existing version of TOME will be available. **"..but don't wait for that. Install TOME today, and create game maps as big as your ideas."** Phil South Amiga Shopper March 1992

AMOS TOME V3.2 costs **£24.99 (£19.99 for AMOS Club Members**) and is only available direct from Shadow Software in the U.K (U.S.A & Australian customers, please contact us for details of more local distributors).

When ordering AMOS TOME, please make cheques payable to Shadow Software.

Send your order to:

Shadow Software, 1 Lower Moor, Whiddon Valley, Barnstaple, N.Devon. EX32 8NW ,England.

All U.K prices are inclusive of V.A.T at 17.5% and all prices include P&P.

....

Please send me AMOS TOME (V3.2)

Name.................... ....

Address................. ....

....

....

Postcode .... **AMOS Club Membership Number** ....

I enclose a Cheque/Postal Order for £19.99 (Members only), £24.99 (Non-Members). If ordering from overseas, please make sure that payment is in British Pounds and drawn on a London Bank. (Eurocheques and U.S Postal Money Orders made out in Pounds are also accepted).

10–Liners

It's a classic card trick and board game this issue, as well as a 10–liner to generate a sprite bank for the 52 cards, 4 counters and a card back !

Card Factory

This program will create the 52 playing cards, 4 counters and card back required for most card and board games and save them to disk.

Othello

This is the classic board game (also known as Reversi), with a simple computer opponent. The object is to capture as many counters on the board as possible. You must place your counters so that you have a counter on either side of one or more of your opponents. This "captures" these counters and changes them to your own. You will need to load in the playing cards created with the Card Factory program first.

Find The Lady

Not a game this time, but a card trick ! Watch as AMOS moves the cards around, and try and spot where the queen will end up. Remember to load in the playing cards created with the Card Factory program first.

Card Factory

```
1) Curs Off : Flash Off : Cls 0 :
 Palette 0 , $ FFF , $ F00 , $ F0 ,
 $ F , $ FF0 , $ 800 , $ 80 , $ 8 ,
 $ 880 : CARD $ = "
 A23456789XJQK " : B = 1 [ 71]
2) For S = 0 To 3 : For C = 1 To
 13 : Ink 1 , 1 : Bar 0 , 0 To 24 ,
 16 : Ink ( S mod 2 ) * 2 , 1 , ( S
 mod 2 ) : C $ = Mid $ ( CARD $
 , C , 1 ) : If C $ = " X " : C $ =
 " 10 " : End If : Text 2 , 8 , C $
 [ 247]
3) Get Icon 1 , 0 , 0 To 25 , 16 :
 Paste Icon - 7 , 16 , Hrev ( Vrev
 ( 1 ) ) : On S + 1 Gosub CLUB ,
 DIAM , SPAD , HEAR : Get Bob
 B , 0 , 0 To 25 , 33 : Inc B :
 Next C : Next S [ 234]
4) For C = 0 To 3 : Cls 0 : For Q
 # = 0 To 2 . 1 * Pi # Step 0 .
 01 : Ink C + 2 : Draw 8 , 8 To 8
 + Cos ( Q # ) * 7 , 8 + Sin ( Q
 # ) * 7 : Plot 8 + Cos ( Q # ) *
 5 , 8 + Sin ( Q # ) * 5 , 6 + C :
 Next Q # : Get Bob B , 0 , 0 To
 16 , 16 : Inc B : Next C [ 75]
5) Ink 2 , 1 , 2 : Set Pattern 2 :
 Bar 0 , 0 To 24 , 32 : Ink 1 : Box
 0 , 0 To 24 , 32 : Box 1 , 1 To
 23 , 31 : Get Bob B , 0 , 0 To
 25 , 33 [ 58]
6) Save " CardPack . Abk " : End
 [ 4]
7) CLUB : Ink 0 , 0 : For A = 0 To
 4 : Draw 12 , 16 To 10 + A , 12
 : Draw 17 , 18 - A To 7 , 14 + A
 : Next A : Draw 12 , 16 To 12 ,
 20 : Return [ 8]
8) SPAD : Ink 0 , 0 : For A = 0 To
 8 : Draw 12 , 12 To 8 + A , 18 :
 Next A : For A = 0 To 1 : Draw 8
 + A , 19 + A To 11 - A , 19 + A
 : Draw 13 + A , 19 + A To 16 -
 A , 19 + A : Next A : Draw 12 ,
 19 To 12 , 22 : Return [ 43]
9) HEAR : Ink 2 , 2 : For A = 0 To
 8 : Draw 12 , 20 To 8 + A , 14 :
 Next A : For A = 0 To 1 : Draw 8
 + A , 13 - A To 11 - A , 13 - A
 : Draw 13 + A , 13 - A To 16 -
 A , 13 - A : Next A : Return [
 54]
10) DIAM : Ink 2 , 2 : For A = 0
 To 11 : Z = A mod 2 : Draw 12
 + A / 2 , 11 + A / 2 - Z To 7 +
 A / 2 , 16 + A / 2 - Z : Next A :
 Return [ 90]
```

Change the Save statement on line 6 to whatever filename and path you want the cards to be saved as.

Othello

```
1) Dim G ( 9 , 9 ) , P ( 1 ) , DX (
 7 ) , DY ( 7 ) : Curs Off : Flash
 Off : Cls 0 : Get Sprite Palette :
 Cls 2 : Ink 0 : For A = 0 To 10 :
 Draw A * 16 , 0 To A * 16 , 160
 : Draw 0 , A * 16 To 160 , A *
 16 : Next A [ 187]
2) D $ = " - 1 0 - 1 - 1 0 - 1 1
 - 1 1 0 1 1 0 1 - 1 1 " : For A
 = 0 To 7 : DX ( A ) = Val ( Mid $
 ( D $ , A * 4 + 1 , 2 ) ) : DY (
 A ) = Val ( Mid $ ( D $ , A * 4 +
 3 , 2 ) ) : Next A : G ( 4 , 4 ) =
 1 : G ( 5 , 5 ) = 1 : G ( 4 , 5 )
 = 2 : G ( 5 , 4 ) = 2 : Paper 2 :
 Pen 3 [ 47]
3) PLACED = 4 : P = 1 : While P (
 0 ) + P ( 1 ) < 99 : P ( 0 ) = 0 :
 P ( 1 ) = 0 : For A = 0 To 9 :
 For B = 0 To 9 : G = G ( A , B )
 : If G : Paste Bob A * 16 , B *
 16 , G + 52 : Inc P ( G - 1 ) :
 End If : Next B : Next A [ 177]
4) Limit Mouse X Hard ( 0 ) , Y
 Hard ( 0 ) To X Hard ( 159 ) , Y
 Hard ( 159 ) : Locate 22 , 1 :
 Print " Red : " ; P ( 0 ) ; " " :
 Locate 22 , 3 : Print " Blue : " ;
 P ( 1 ) ; " " : Locate 30 , P * 2
 - 1 : Print " < - " : Locate 30 , (
 3 - P ) * 2 - 1 : Print " " [ 164]
5) If P = 1 : OK = 0 : While OK =
 0 : While Mouse Key = 0 : Wend
 : X = X Screen ( X Mouse ) / 16
 : Y = Y Screen ( Y Mouse ) / 16
 : S = 0 : Gosub TEST : If G ( X ,
 Y ) = 0 and FA = 0 : G ( X , Y )
 = P : OK = 1 : End If : Wend [
 165]
6) Else For S = 9 To 1 Step - 1 :
 For X = 0 To 9 : For Y = 0 To 9
 : If G ( X , Y ) = 0 : Gosub TEST
 : If Z > = S and Rnd ( 6 ) : G (
 X , Y ) = P : S = 0 : Gosub
 TEST : X = 99 : Y = 99 : S = - 1
 : End If : End If : Next Y : Next
 X : Next S : End If [ 91]
7) Inc PLACED : P = 3 - P : Wend
 : Locate 22 , 10 : If P ( 0 ) > P (
 1 ) : Print " You Won ! " : Else
 Print " Computer Won ! " : End If
 : End [ 56]
8) TEST : Z = 0 : FA = 1 : For D
 = 0 To 7 : H = DX ( D ) : V =
 DY ( D ) : E = 0 : If X + H > = 0
 and X + H < 10 and Y + V > = 0
 and Y + V < 10 : E = G ( X + H
 , Y + V ) : End If : If E = 3 - P :
 C = 2 : F = 0 : F2 = 0 : While F
 = 0 and F2 = 0 and C < 10 [ 197]
9) G = 0 : If X + H * C > = 0 and
 X + H * C < 10 and Y + V * C >
 = 0 and Y + V * C < 10 : G = G
 ( X + H * C , Y + V * C ) : End
 If [ 90]
10) F = ( G = 0 ) : F2 = - C * ( G
 = P ) : Inc C : Wend : If F2 and
 F = 0 : Z = Z + F2 - 1 : FA = 0
 : If S = 0 : For A = 1 To F2 : G
```

```
( X + H * A , Y + V * A ) = P : Next A : End If : End If : End If : Next D : Return [ 66]
```

Find The Lady

```
1) Timer = 0 : N = 52 : Curs Off : Flash Off : Get Sprite Palette : C = 1 [ 78]
2) Cls 0 : Paper 0 : Pen 3 : Home : Print " Wow ! " : For A = 0 To 31 : For B = 0 To 7 : If Point ( A , B ) = 3 : Paste Bob A * 8 + 16 , B * 16 + 8 , C : Add C , 1 , 1 To 52 : End If : Next B : Next A [ 38]
3) Locate 0 , 20 : Pen 2 : Centre " Press a key to play " : Wait Key : C = 0 [ 48]
4) Do : Cls 0 : Home : Print " Look , Nothing up my disk drives ! " : Wait Key : Home : Cls 0 : Print " I place 3 cards down on the table . . " : For A = 0 To 2 : Bob A + 1 , A * 64 + 64 , 100 , A + 50 : Wait 50 : Next A : Wait Key [ 170]
5) For A = 0 To 2 : Bob A + 1 , 64 + A * 64 , 100 , 57 : Next A : M $ = " ; M 0 , - 64 , 2 ; M 64 , 0 , 2 ; M 0 , 64 , 2 " : M2 $ = " M 0 , 64 , 2 ; M - 64 , 0 , 2 ; M 0 , - 64 , 2 " [ 151]
6) For A = 0 To 2 : Channel A + 1 To Bob A + 1 : Next A : Cls 0 : Home : Print " Press a key to shuffle them ! " : Wait Key [ 56]
7) For B = 0 To 5 : A = ( B + Rnd ( 1 ) ) mod 2 : Amal 1 + A , M $ : Amal 2 + A , M2 $ : Amal On : Wait 14 : For A = 0 To 2 : Bob 1 + A , 64 + A * 64 , 100 , 57 : Next A : Next B [ 237]
8) Cls 0 : Home : Print " Click on the Queen ! " : OK = 0 : While OK = 0 : While Mouse Key = 0 : Wend : OK = Max ( 0 , Min ( 2 , ( X Screen ( X Mouse ) - 64 ) / 64 ) ) + 1 : Wend [ 37]
9) For A = 0 To 2 : Bob 1 + A , 64 * A + 64 , 100 , 57 : Next A : B = Rnd ( 100 ) : C = - ( B > 95 ) - ( B < 96 ) * Rnd ( 1 ) * 2 : Bob OK , ( OK - 1 ) * 64 + 64 , 100 , 50 + C : Cls 0 : Home [ 252]
10) If C = 1 : Print " You got me , You win ! " : Else Print " Wrong ! Try Again . . . " : End If : Wait Key : Bob Off : Loop [ 117]
```

Remember, that these 10 liners and a few others are on this issue of THE DISK, see page 8 for details.

Mini Maths 7

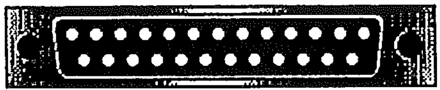
Justifying Scores

Most games show the score with leading zeros, e.g 0000240 . This effect can be acheived quite simply in AMOS using string functions. If the variable **SCORE** holds the score, you could do:

```
S$=Str$(SCORE) : Rem turn it into a string
S$=Mid$(S$,2) : Rem remove the leading space
S$=Right$("0000000"+S$,7) : Rem add leading zeros
```

S$ can now be printed, text'ed or Ctext'ed onto the screen .

Serious Serial



Just about everyone now has the Serial Extension installed on their AMOS set up now, the only versions of AMOS without it are pre V1.23 (Get upgraded now !).

Unfortunately, only those of you who got the Compiler, or remembered to keep the document files from the V1.23 update have actually got any instructions for the extension, and due to a couple of "features" (I.e whopping great bugs !) hardly anyone has been able to use it !

Part 1. Bug Fixing ! The major problem was this, you do a **Serial Open** command in your program, followed eventually by a **Serial Close** command to finish. Run the program once, no problem, try to run it again and the Amiga goes into infinite think mode (I.e it crashes horrendously). This was caused by a slight bug in the older Workbench **Serial.Device.** So on the V1.23 update disk, Francois kindly included a newer serial.device (we suspect from Workbench 2 !) that solves the problem. All you have to do is copy it across to the Devs directory of your boot disk and all is hunky dory. However, when AMOS 1.3 was released (the Compiler version), certain people at Mandarin/Europress forgot about the serial bug and put the OLD serial.device on the updater disk, the same happened with V1.32 ! End result lots of people finding out that the serial extension doesn't work again !

Fortunately, the new version of AMOS 1.34 (and **THE DISK**) now has the correct **Serial.Device** on its disk. So if you get a horrendous crash from your serial open, make sure you have the correct file copied across. The correct serial.device is **5292** bytes long.

Part 2. Usage First of all, you have to open up the serial port to send or receive anything. For this you use the Serial Open command, its format:

SERIAL OPEN Channel , Serial [, shared , xdisabled , 7wires]

Channel is the Channel number (similar to files) which can be from 0-3

Serial is the logical number of the device to access if you are using a Multi Serial card. Normally, you would just bung a zero here.

The **Shared** flag tells the Amiga whether or not AMOS wants to allow the serial channel to be shared with other tasks. If set to 0 (the default) your program will be the only one using the port. If set to -1, other programs can be sending and receiving at the same time as your one. This can be tricky !

The **Xdisabled** flag must be set to true (-1) if you want to use **XON/XOFF** checking in your communications. The **Serial X** command must be used to set the XON/XOFF characters.

The **7Wires** flag (again 0 or -1) sets the port into 7wires mode. Normally you would either set this to 0.

Normally, say for comms between 2 Amigas directly linked with a null modem cable, you just use **SERIAL OPEN 0 , 0**

When you have finished using the serial port, you can close it with **SERIAL CLOSE channel**

You might (I.e 99% probably) want to change the speed of the communications to another baud rate. For this, you can use:

```
SERIAL SPEED Channel , baud
```

This command will reject any non-available baud rates. For normal Amiga Link comms, **9600 baud** is quite reasonable.

Now you will want to know how to send data. There are 2 commands for this:

```
Serial Send Channel , String$
```

This will send a string of characters over the serial port. AMOS doesn't hang around waiting for it to finish, so you will need to use **Serial Check** to make sure the device is ready for your next send. e.g

```
A$ = " Serial Communication "
Serial Send 0 , A$
While Serial Check ( 0 ) = 0
Wend
A$ = " Is wonderfull ! "
Serial Send 0 , A$
```

You can also use

```
SERIAL OUT Channel,address,length
```

which sends raw data, starting at address and length bytes long. Again, use Serial Check to make sure it is finished.

```
= SERIAL CHECK ( Channel )
```

returns False (0) if the channel isn't ready and True(-1) if it feels up to sending or receiving.

To get data from the serial port, you have....

```
= SERIAL GET ( Channel )
```

which will return -1 if there was nothing here, or one byte if there was anything waiting. You also have

```
= SERIAL INPUT$ ( channel )
```

which will wait for a string (ending in Linefeed and Carriage return) Which is ine if you are sending messages, but otally useless for MIDI or data transmission. Normally to receive a message, you would do..

```
A = - 1
While A = - 1
A = Serial Get ( 0 )
Wend
M$ = Chr$ ( A ) + Serial Input$ ( 0
)
```

Use with MIDI.. If you are using MIDI, you can just about get away with using the Serial Extension to handle it. I'm working on a specially modified serial extension to handle MIDI more easily, but for the moment, here's what you have to do.

First use the **SERIAL FAST Channel** command to set the serial port into high speed mode. Then use the **SERIAL BUFFER** command to set the buffer to a higher value, say 5000. So you would do **SERIAL BUFFER 0,5000** (there's hardly any synth that sends over 5000 bytes in a single message without acknowledgement).

Then send your data from banks, using **SERIAL OUT** and receive it using **=SERIAL GET()**. This will be quite slow, although not too bad, and because the buffer has been increased there will be no worry over data loss.

The MIDI commands I'm bolting on will basically form a new Serial Extension, which I'm going to be generous and release onto P.D (Having sent the source code to Francois to include in AMOS 2). So keep communicating !

LaserDrome

Any AMOS Club members living in or near Peckham in London, who like running around and shooting people with real Lasers, (sort of like paintball but less painful) have a go of the LaserDrome at 267 Rye Lane. I had a go while in London and it's excellent fun ! (Not to mention a **LOT** cheaper than most of the arcades in London)

Licenseware Reviewed

LPD 69 Magical Mix Up by Daisy Software.

"Oh goody, another kiddies game !" I thought upon seing the title. One hour later I had to go and have a stiff drink !

Despite being an exellent game for children, this one can get difficult ! The game is very simple, you have to take the cut of blocks of a picture and put them in the correct order into a frame (like a simple jigsaw). However, you have a time limit, and the further you get into the game, the more pieces each picture is cut into. When the game gets to the 4 pixel high horizontal strips stage it makes great practise for Government investigators who need to learn how to un-shred shredded documents !

A few points could be better, for instance there should be a little more incentive to get further, and the scoring system is a little suspect, but the gameplay is about right for this type of game. **8 / 10**

LPD 79 Music Engine by Paul Townshend

The basic idea of this disk is to provide you with play routines that you can bolt onto AMOS, so that you can use Sonix, Delta Music, OctaMed, Soundtracker etc music files without having to convert them. This is a nice idea, but it's a shame that this sort of disk is required, as the AMOS music bank setup has more potential power than most of the "Hacker" systems, and with a proper editor (not a hacker system, or score only editor) AMOS users wouldn't have to muck about with installing play routines. The disk is good as a temporary fix however, and if you are desperate to us Sonix files in your programs, try it. **8 ,/ 1 0**

All the Licenseware and official AMOS P.D Software can be obtained from:

Deja Vu Software, 25 Park Road
Wigan, WN6 7AA. ☎ 0942 495 261

The official AMOS P.D distributors.

Book Report. Still no news on a firm release date for Stephen Hill's Games Makers Manual. It was intended to be released in January, but there seems to have been a problem with converting diagrams and Sigma Press have yet to set another release date.

The only info we have is that the book is 400-450 pages, has Chapters on most aspects of AMOS Games programming including The Compiler and AMOS TOME and will be in the £16-20 price bracket. We'll keep you posted.

Another new book about to be released is Len Tucker's AMOS IN ACTION, which teaches you how to program games by taking you through how Len wrote his game Mervin The Martian (in Licenseware). This book details various sorts of programming technique and how to use the various utilities in AMOS to write a game (including the Compiler, TOME, Sprite X and CTEXT).

AMOS IN ACTION comes with a sourcecode disk (saves you typing) and should be out Mid to late March.

This Newsletter was written by Aaron Fothergill, unless otherwise stated, put together on an Amiga and an ST using Timeworks DTP .

Graphics by Adam Fothergill.

Printed by Creative Printers, 104 Boutport Street, Barnstaple, N.Devon. Telephone (0271) 77655

Special Offers

Because of problems with the post over Christmas, many of you didn't receive your issue 5/6 newsletters in time to take advantage of the special offers for resubscription. Because of this, all the offers are now valid until Volume 2, Issue 2 is released (End of April). Also the Competitions run in the last issue are being run again, so that those of you who missed the deadline have a chance of winning.

o The Special Offers

If you order AMOS TOME (at the AMOS Club member's price of £19.99) then you get the **AMOS TOME Goodies Disk 1 free** .

Also, If you haven't already got the excellent Sprite X and Ctext packages, then you can order them for just 2 pounds each (saving 50p on the normal member's price). Postage and packing is included (even for overseas orders).

This form is also the one used to order **THE DISK** (See Page 8)

o Win Lots of Software !

All resubscription forms that get to us between January 1st and the end of April 1992, will be put in a large hat (well, a large bag) and the first drawn out will win a complete AMOS developer's kit, containing:-

The AMOS Compiler (supplied by Europress Software), AMOS TOME V3.2, Sprite X V1.32, CTEXT (Supplied by Shadow Software), Rainbow Warrior (By Martin Brown, Supplied by Deja Vu P.D) and 10 P.D Disks of the winner's choice (Also supplied by Deja Vu P.D).

2 runners up can choose The AMOS Compiler or AMOS TOME, and there are 4 more prizes of Sprite X and Ctext for the next 4 out of the bag.

Winners will be anounced in the next newsletter (Volume 2, issue 2)

....

I wish to order some goodies

Name.......................................**Membership.Number**. ....

Address................................ ....

....

....**PostCode**...

I enclose ☐**£10.00 for Back Issues 1-6**

☐**£19.99 for AMOS TOME (inc free goodies disk)**

☐**£7.00 for a years subscription to THE DISK (£10 Overseas)**

☐**£2.00 for AMOS Sprite X V1.32**

☐**£2.00 for AMOS CTEXT**

☐**£1.75 for THE DISK , issue1 (£2 Overseas)**

Total £.................................

When ordering from overseas , please make sure that payments are in British Pounds and are drawn on a London bank (Eurocheques and U.S Postal Money orders made out in Pounds are also acceptable)